

```
// COMPARAR FECHAS – DOUBLE LINKED LIST
```

```
@Override
```

```
public int compareTo(Reserva reserva){  
    return this.fechaReserva.compareTo(reserva.getFechaReserva());  
}
```

```
// LIBROS ESCRITOS COMPARACION
```

```
public int compareTo(LibrosEscritos libroEscrito){  
    return Integer.compare(this.cantidad, libroEscrito.getCantidad());  
}
```

```
// CONTAR LIBROS
```

```
public static DoubleLinkedList <LibrosEscritos> librosPorAutor(DoubleLinkedList <Libro> libros){  
    DoubleLinkedList <LibrosEscritos> librosAutores = new DoubleLinkedList<LibrosEscritos>();  
    for (Libro libro : libros){  
        String autor = libro.getAutor(); // Se actualiza el nombre del autor de la lista original  
        LibrosEscritos existente = null; // Se declara un elemento a llenar para comprobar si el autor ya existe en la nueva lista  
        for (LibrosEscritos librosEscritos : librosAutores){  
            if (librosEscritos.getName().equals(autor)){  
                existente = librosEscritos; // Existente pasa a tener los valores del primer libro cargado  
                break; // Si lo encuentra deja de recorrer la lista  
            }  
        }  
        if (existente != null){ // Significa que el autor ya existe, por lo tanto,  
            existente.setCantidad(existente.getCantidad() + 1); // Se incrementa su cantidad de libros  
        }else{ // Caso contrario  
            librosAutores.AddInOrder(new LibrosEscritos(autor,1)); // Se lo agrega a la lista con 1 como cantidad inicial  
        }  
    }  
    return librosAutores;  
}
```

```
import Trabajo_PracticoN6.Ejercicio8.SimpleLinkedList;  
public class Pila<Element> {  
    private SimpleLinkedList<Element> list;  
  
    public Pila() {  
        this.list = new SimpleLinkedList<Element>();  
    }  
  
    public void push(Element element) {  
        this.list.addFirst(element);  
    }  
}
```

```
public Element pop() {
    if (this.empty()) {
        throw new RuntimeException("La pila esta vacía...");
    }
    return this.list.removeFirst();
}

public Element peek() {
    if (this.empty()) {
        throw new RuntimeException("La pila esta vacía...");
    }
    return this.list.getFirst();
}

public boolean empty() {
    return this.list.size() == 0;
}

public int size() {
    return this.list.size();
}

@Override
public String toString() {
    return list.toString();// llama al toString de la lista
}

}
```